Assignment 2: First Iteration

Harry Potter: The Compiler of Secrets

This assignment follows on from Assignment 1. In that assignment, you designed some extensions to the functionality of a rogue-like game set in the Harry Potter universe. In Assignment 2, you will implement those extensions.

You may change your design if you find that you need to. It is normal for the design to evolve alongside the codebase as the developers' understanding of the requirements improves. If your design changes, you must document the changes and explain the reasons for them in an updated Design Rationale document.

In Assignment 3, you will be given further requirements to design and implement, as a well as a written task, regarding recommendations for improvements to the codebase.

Project requirements

All the functional requirements stated in the Assignment 1 specification apply.

Design is important

One of the primary aims of this unit is for you to learn the fundamentals of object-oriented design. In order to get a high mark in this assignment, it will not be sufficient for your code to "work". It must also adhere to the design principles covered in lectures, and in the required readings on Moodle.

Updating your design

As you implement your design, you may find that it has problems. You might also think of a better approach to some of the problems you've faced — your understanding of the problem will improve as you progress. If you want to update your design, you may do so; if you decide to do this, be sure to update your design documents so that they match the code, and write a brief explanation of your changes and the reasons behind them. This will help your marker understand the thinking behind your code.

Coding and commenting standards

You must adhere to the Java coding standards that were posted on Moodle earlier in the semester.

Write javadoc comments for *at least* all public methods and attributes in your classes.

You will be marked on your adherence to the standards, javadoc, and general commenting guidelines that were posted to Moodle earlier in the semester.

Bonus marks

There are bonus marks available for groups that come up with ideas for new features for the game, and then design and implement them. These could be new kinds of actor or entity, with new behaviours. They could be new kinds of location. They must be different from any features that have been requirements in FIT2099 in previous semesters.

A feature must be quite complex to qualify for a bonus mark. You must discuss your plans for bonus mark features with your tutor and gain approval before beginning work on them. Bonus marks will be awarded after the submission of Assignment 3, but can be started now.

A maximum of three bonus marks are available (these are whole marks for the unit). No one has to attempt bonus marks, and it is possible to get full marks for the unit without any bonus marks.

Submission instructions

The due date for this assignment is *Tuesday 29th January at 23:59*. We will mark your Assignment 2 on the state of the "master" branch of your Monash GitLab repository at that time. If you've done any work in other branches, make sure you merge it into master before the due time.

Do not create a new copy of your work for Assignment 2. Continue working on the same files, in the same directory structure (you might like to add a tag to the your final Assignment 1 commit before starting on Assignment 2, so you can find that version easily).[1]

As we said above, you may update your design if you find that your initial version is unworkable, or if you think of a better approach during implementation. You may also update your Work Breakdown Agreement if you need to, but make sure the other members of your team accept it. If you do either of these things, please make sure you say so. Your README.md file is a good place to put these sorts of release notes.

Unless a team member has applied for and received special consideration according to the Monash Special Consideration Policy,[2] late submissions will be penalized at 10% per day late.

It is both team members' responsibility to ensure that the correct versions of the documentation and code are present in the repository by the due date and time. Once both teammates have agreed on a final Assignment 2 submission, do not make further commits to the master branch of the repository until the due date has passed, without the agreement of your teammate. If you want to continue to make changes to the repository for some reason, make another branch.

We will take your Work Breakdown Agreement into account when marking if there seems to be a major discrepancy in the quality of different parts of the submission, or if the code is missing major sections. Students whose work is inadequate in either quality or quantity will be penalized, and their partners will be compensated. If you choose to reallocate tasks, make sure you keep your WBA up to date.

Getting Marking

You must explain your submitted documents to your tutor during your interview. Each team will be allocated a 15 minute slot in which to be interviewed in the week of the deadline. Both team members must explain the artefacts they have produced, and the rationale behind them.

The interview is a hurdle for getting marked.

Marking Criteria

This assignment will be marked on:

- Functional completeness

- Adherence to design principles

- Code quality

    - readability

    - adherence to Java coding standards

    - quality of comments

    - maintainability (application of the principles discussed in lectures)

- Quality of updated documentation, if any

- Correct use of GitLab

Marks may also be **deducted** for:

---

[1] https://git-scm.com/book/en/v2/Git-Basics-Tagging
[2] http://www.monash.edu/exams/changes/special-consideration

- late submission

- inadequate individual contribution to the project

- academic integrity breaches